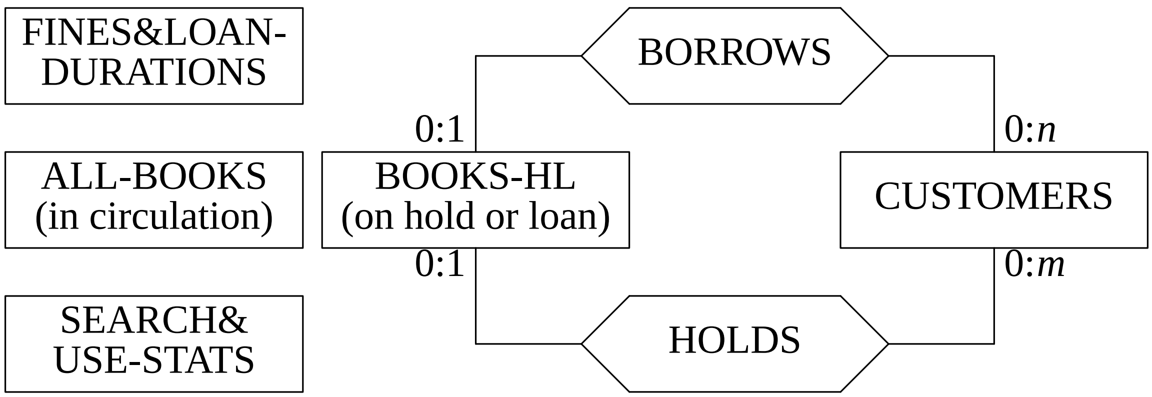
FINITE-STATE MODEL, DATAFLOW MODEL, AND ENTITY-RELATIONSHIP MODEL

FOR

BUILDING AND ANALYZING

REQUIREMENTS

ENTITY-RELATIONSHIP DATA-MODEL



0:*n* on CUSTOMERS to BORROWS link:

A customer may at any time have a minimum of 0 books and a maximum of *n* books borrowed.

0:1 on BOOKS-HL to BORROWS link:

A book may at any time be borrowed by a minimum of 0 and a maximum of 1 customer.

0:*m* on CUSTOMERS to HOLDS link:

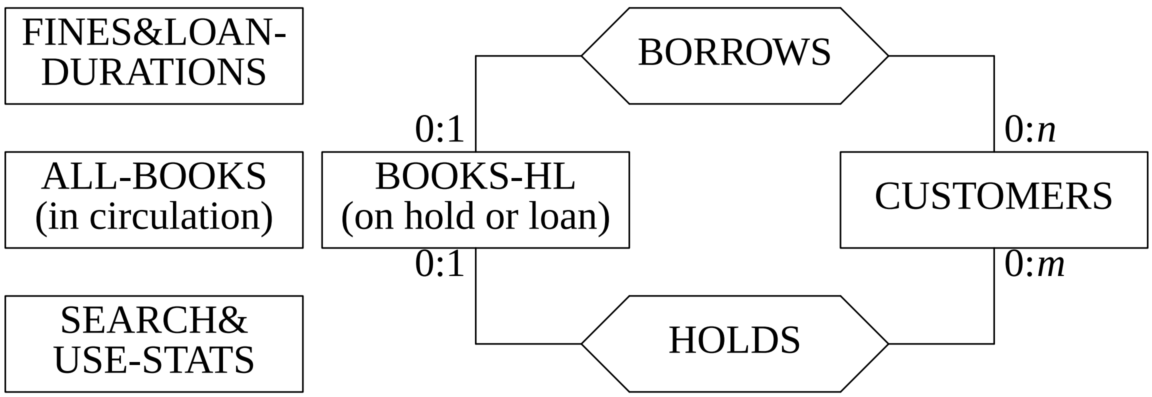
A customer may at any time have put hold on a minimum of 0 books and a maximum of *m* books

0:1 on BOOKS-HL to HOLDS link:

A book may at any time have hold by a minimum of 0 and a maximum of 1 customer.

- The books in BOOKS-HL have either been borrowed or have a hold, thus minimum cardinality 0 cannot be true at the same time.

ER-MODEL FOR AT MOST ONE HOLD(Contd.)



ALL-BOOKS:
(*BookId*, BookType, Title, Author, Publisher, PurchaseDate, PurchasePrice)

SEARCH & USE-STATISTICS:
(*BookId*, SearchCount, BorrowCount, TotalUseDuration)

FINES & LOAN-DURATIONS:
(*BookType*, *CustomerType*, LoanDuration, LoanRenewalDurarion, HoldingPeriod, Fine, ReplacementCostPolicy)

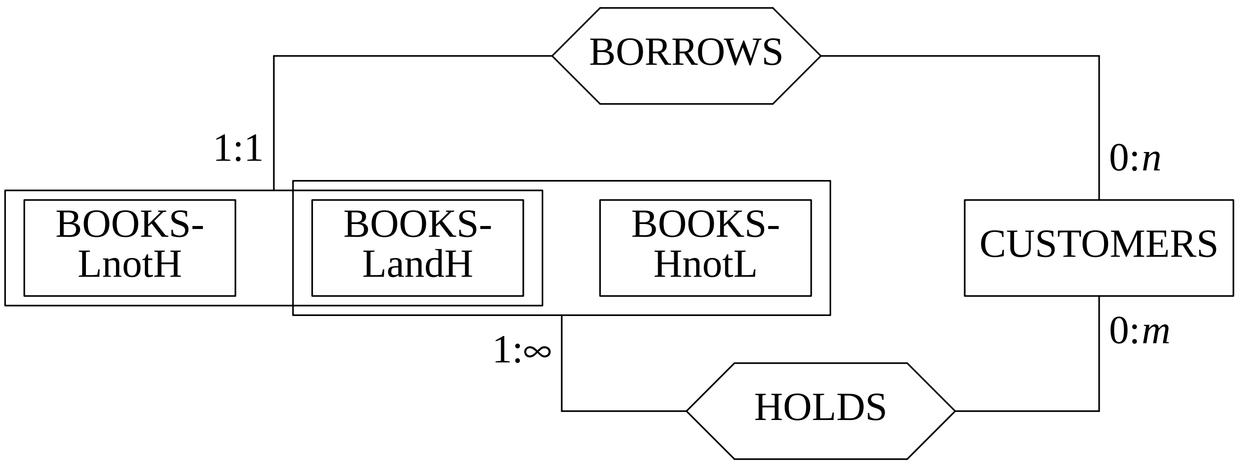
BOOKS-HL: (*BookId*, BookType, HasHold)

CUSTOMERS: (*CustomerId*, CustomerType, Address, TotalBorrowCount, TotalHoldCount, TotalLateRetCount, TotalLostBookCount)

BORROWS: (*BookId*, CustomerId, LoneDate, ReturnDate, DueDate, ReminderCount)

HOLDS: (*BookId*, *CustomerId*, HoldReqDate, HeldStartDate, HeldEndDate)

A REFINEMENT OF BOOKS-HL ENTITY

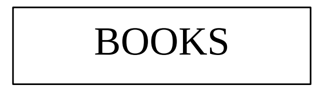


- We are also allowing here multiple holds, one for each different customers (not shown in the model itself).
- The fact that the same person currently borrowing a book cannot put a hold on it is not shown in the model either.

ENTITY vs. AN ATTRIBUTE

- An attribute stands only in the context of an entity, not by itself.
- An entity can have just one attribute.

Representing an Entity Using a Relationship:



BOOKS: (*BookId*, BookType, Title, Author, Publisher, PurchDate)

BookId	BookType	Title	Author	Publ.	PurchDate
b#1	tp1	tt1	a1	pb1	pd1
b#2	tp1	tt2	a1	pb1	pd2
b#3	tp3	tt2	a3	pb3	pd3

BOOKS′: (*BookId*, Title, Author, Publisher, PurchaseDate)
BOOK-TYPES: (*BookType*, LoanDuration)
HAS-TYPE: (*BookId*, BookType)

Bk-Type	Loan-Dura.
tp#1	d1
tp#2	d1
tp#3	d3
tp#4	d4

BkId	Bk-Type
b#1	tp1
b#2	tp1
b#3	tp3

BkId	Title	Author	···
b#1	tt1	a1	···
b#2	tt2	a1	···
b#3	tt1	a3	···

Question: What is the advantage in the new ER-model?

EXERCISE

1. Table 10.1 in the text book lists the following ER modeling concepts.

Entity	distinguishable object of some type
Entity type	type of a set of elements
Attribute value	piece of information describing an entity
Attribute	type of a set of attribute values
Relationship	association between two or more entities

   Why can't we talk about relationship types and relationship values? Give examples to explain your answer. What are some the relationship values for HAS-TYPE relationship in the example on the previous page?

2. When we represent an entity using a relationship, we always get one of the cardinalities as "1:1". Is the converse true, i.e., if a relationship cardinality is "1:1" then can we replace the relationship and build an entity somehow in its place? Explain with an example.

3. What are some key differences between ER-modeling and ER-diagram?

4. ER-modeling tells only part of the story - which part and which part is not captured?

DATAFLOW DIAGRAMS: AN ALTERNATIVE SEMI-STATIC MODELING TOOL

- Combines some of FSM's dynamic features with ER's static features; shows action's input-output.
- Standard DFD cannot model conditions for actions; not suitable for low-level modeling.

Structure:

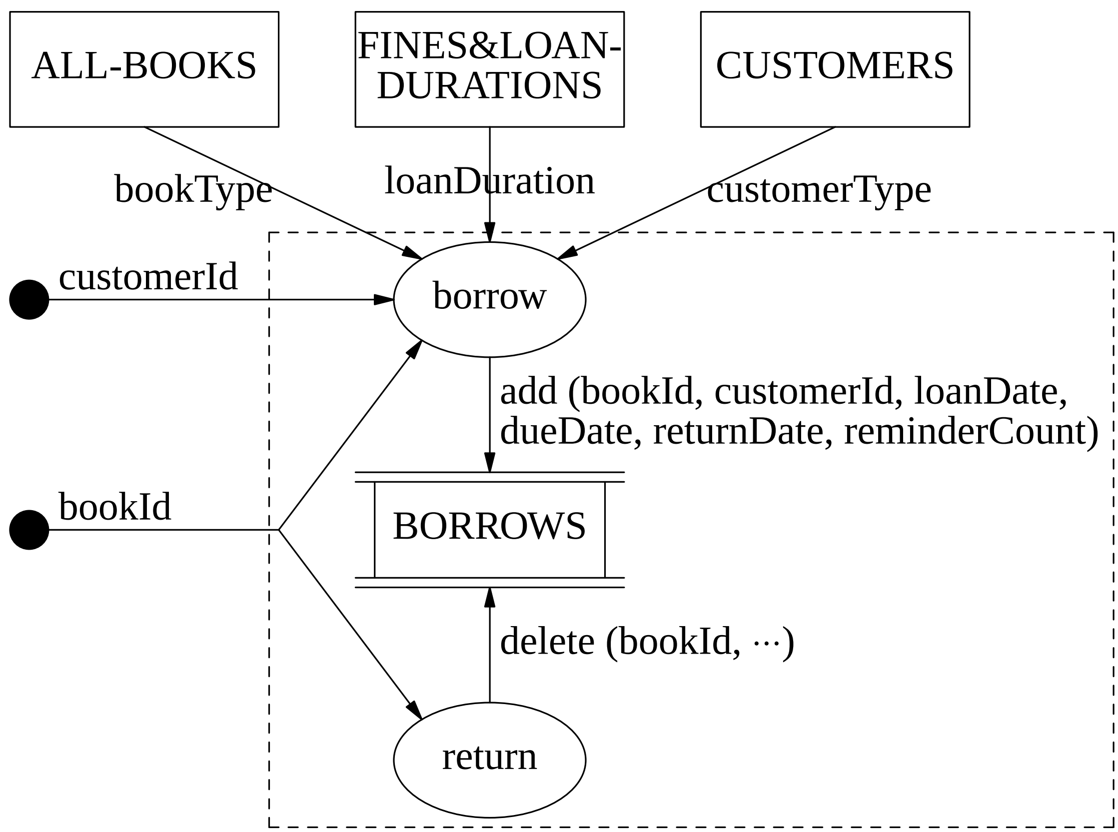
- Has 4+1 = 5 types of nodes:
  - Two external node types: data-sources and data-sinks. (In reality, they can also be processes.)
  - Two internal node types: Process-nodes and data-store nodes. (The data-stores can act as data-source and data-sink.)
  - Special nodes to represent parameters to operations, if any.
- There are links connecting two process-nodes or a process-node and a data-node.
  - Links represent data-flows.

Semantics:

- Process: a major function (it may be implemented by many smaller functions).
- Data-store: an interface between asynchronous processes, which do not have a calling relationship between them.
- Links: inputs and outputs of processes.

DATAFLOW EXAMPLE: BORROW AND RETURN OPERATIONS

DFD for *Successful* Borrow and Return Operations:



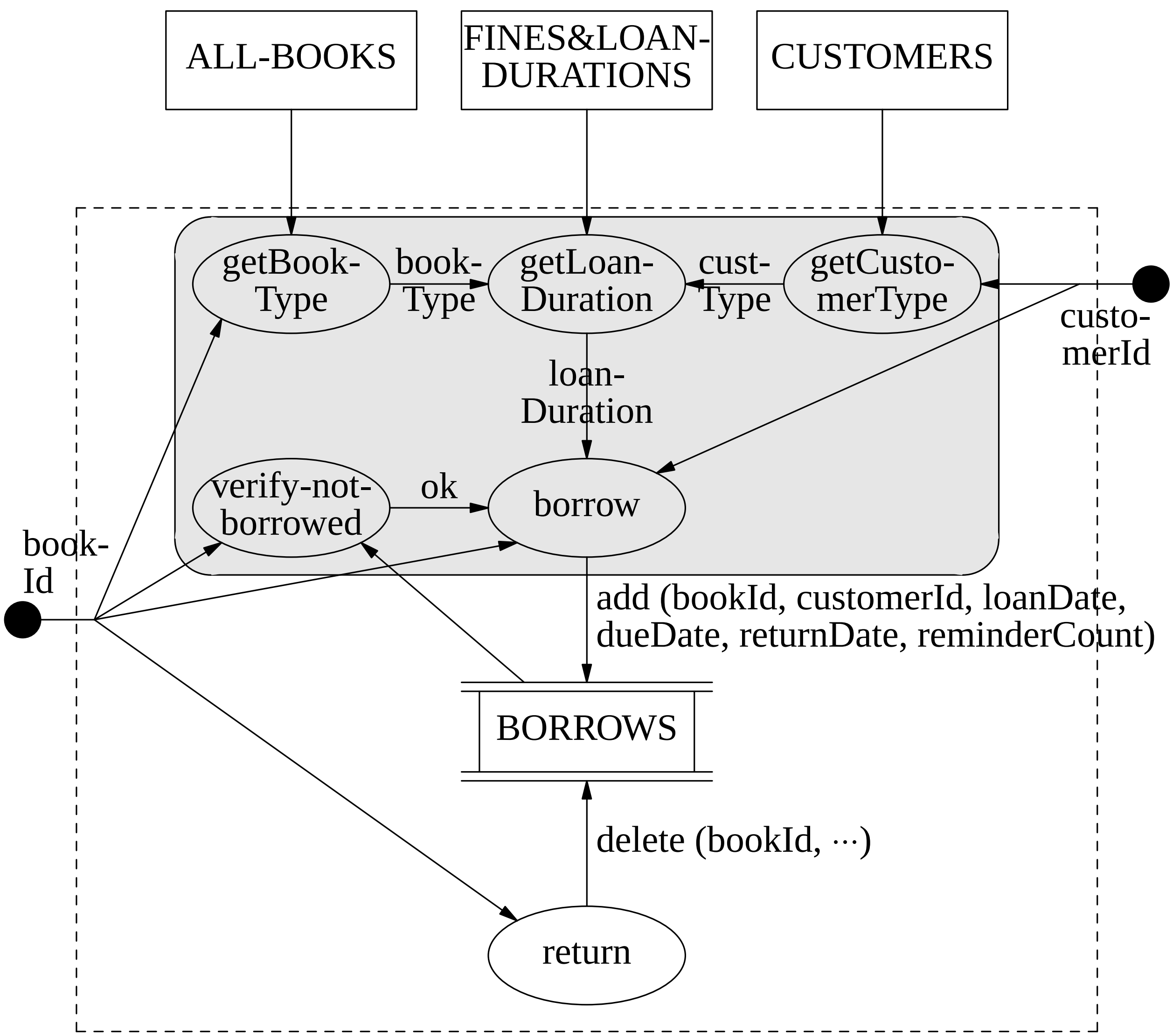
Question: Do you see any missing dataflow for borrow-operation?

EXERCISE

1. Modify the above dataflow diagram by breaking down the borrow-operation into several suitable smaller operations.
2. Also, add guards to model the fact that a book can be borrowed by at most one customer at any time. Recall that BORROWS holds only the current borrow-information.

EXPANDED DATAFLOW FOR BORROW AND RETURN OPERATIONS

DFD for *Successful* Borrow and Return Operations:



Question: Why should we make the data-store BORROWS internal? Would it still be internal if we are only modeling the borrrow-operation?

CONSTRAINTS ON LINKS AND LABELS

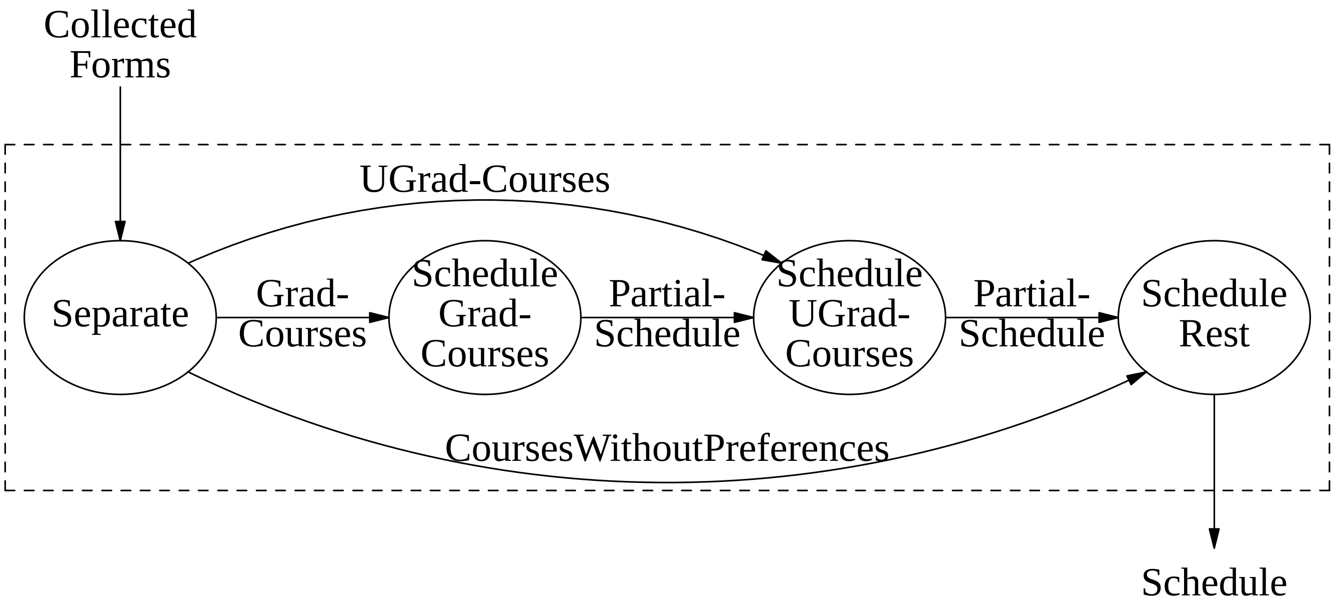
- Labels on links and nodes:
  - Each link has a data-label (noun) as is each data-nodes (internal or external).
  - Each process-node has an action (verb) label.
  - Each process-node has ≥ 1 links (inputs) to it and ≥ 1 links (outputs) from it. (Few exceptions: random number generators has no incoming link and function to free memory has no outgoing links.)
  - No two links to a process has a common label; two links from a process may have a common label.
  - Each data-store node has at least a link (inputs) to it or a link (outputs) from it.

    All links to/from a data-store have the same label (except when the data-store name is "d_1 or d_2", in which case the link label can be "d_1", "d_2", or "d_1, d_2").
- Paths and connectivity
  - Each process-node must be on a path from a external data-source to a data-sink (save the exceptions).
  - The internal nodes should form a connected graph (as an undirected graph).

Question: How many ways can a diagram fail to be a valid dataflow diagram?

A NOT-SO-GOOD DATAFLOW MODEL

DFD with Four Operations:†



Questions:

- ? What structural problems do you see in this dataflow model?
- ? What assumptions have been made here about grad-courses vs. undergrad-courses?
- ? What dataflow (label) problems do you see? Are there better ways of labeling some of the dataflows and processes?
- ? Show the new DFD if we do not separate scheduling of grad-courses and undergad-courses?
- ? Are there missing data-sources and data-sinks?
- ? Show the new DFD if we assume that there can be scheduling-conflicts for courses with schedule-preferences.

† Example from page 151 in Software Engg, by P. Jalote.